operating delivery technologies such as, for example, cellular phones, palm pilots, pagers or any other wireless mechanism for request, receipt and display of information. The presentation format for the WML page 38 provides communication with the delivery technologies using website Meta language.

In other embodiments, the end-user systems layer 12 may include other types of delivery technologies such as, for example, satellite based communication devices, automated systems and devices, such as automated teller machines, or any other type of communication device. In addition, the front-end systems layer 14 may include other types of presentation formats such as, for example, unformatted text, Directory Service Markup Language (DSML), proprietary formats, such as Microsoft™ Word, or any other presentation format. Further, delivery technologies within the end-user systems layer 12 may be communicatively coupled with the different presentation formats within the front-end systems layer 14 in configurations other than those illustrated in FIG. 2. For example, the B2B user 24 may be communicatively coupled with the HTML page 32 and the web user 22 may be communicatively coupled with the XML page 34.

Referring again to FIG. 2, the illustrated embodiment of the business services layer 16 includes core classes represented by ApiService class 42, Message class 44, Field class 46 and BusinessService class 48. In addition, the business services layer 16 includes MESSAGEDEFINITION class 50 and an extensible stylesheet language (XSL) script 52. In other embodiments, fewer or more classes may be used to provide the functionality of the business services layer 16 described herein. The embodiments of the business services layer 16 are preferably implemented using Java technology. In other embodiments, implementation may be with other object-oriented technologies, such as, for example C, C++ or Microsoft™ C sharp (C##).

Java technology is a well-known object-oriented technology. Objects within object-oriented technologies include a state maintained in one or more variables. Behavior of objects is implement with methods, which are functions (subroutines) associated with the object. A particular object is called an instance. An instance of an object is instantiated (or created) by a constructor. Multiple objects of the same kind may be part of a class or a subclass within the class. Objects within one or more classes form a program or application through interaction with each other using

messages. Messages contain parameters that are information indicating how an object is to perform a corresponding method within the program.

Programs created with Java technology use a Java programming language operating on a Java Virtual Machine (Java VM). The Java VM may be ported onto various hardware-based platforms that include an operating system and hardware. Exemplary hardware-based platforms include Window NT™, Windows 2000™, Linux™, Solaris™ or MacOS™ operating on a computer.

The Java programming language runs on any implementation of the Java VM. The Java programming language source code for one embodiment of the business services layer 16 is included in the computer program listing appendix filed herewith.

The Java technology also includes Java Application Programming Interface (API). Java API is a large collection of ready-made software components providing a wide range of functionality. The software components provide "off the shelf" capability that may be implemented within Java based programs. As discussed herein, software components that are from Java API are designated by a name followed by "API."

Programs written in the Java programming language may be characterized as applets, servlets and applications. Applets are programs that adhere to certain conventions allowing the applet to run within a Java-enabled browser. Applications are standalone programs running directly on a Java platform that includes the Java VM and Java API. Some applications operate as servers, such as, for example, Web servers, mail servers and print servers to serve clients on a network. Servlets are run time extensions of applications that operate within Java Web servers to configure or tailor the server.

Referring again to FIG. 2, ApiService class 42 is preferably a servlet directing the overall operation of the business services layer 16. In one embodiment, Java servlet technology provides a mechanism for implementing the request/reply mechanism for ApiService class 42. In other embodiments, analogous implementations using Enterprise Java Beans, JavaServer™ Pages (JSP) or Microsoft™ Application Server Pages (ASP) may be used.

ApiService class 42 is a generic mechanism for executing custom application code within subclasses of BusinessService class 48. The custom application code

may be executed as a function of requests received from the front-end systems layer 14.

The requests received by ApiService class 42 of one embodiment may be in a Servlet Request Format. The Servlet Request Format may be a generic servlet format or may be implemented as, for example, an HTTP servlet or any other format. The presentation formats included in the front-end system layer 14 may be used to create requests in the Servlet Request Format. In other embodiments, the requests may be in other presentation formats such as, for example, unformatted text, HTML, WML, DSML, proprietary languages such as, Microsoft™ Word or any other language. In addition, ApiService class 42 provides output messages based on responses by the subclasses of BusinessService class 48. In one embodiment, the output messages may be provided as XML, HTML or WML. In other embodiments, the output messages may be in other presentation formats such as, for example, unformatted text, DSML, proprietary languages such as, Microsoft™ Word or any other language.

Languages such as XML may be utilized to represent self-describing data structures. Self-describing data structures identify units of data, referred to as an element, using tags. Tags are collectively referred to as markup. XML allows the definition of the tags to provide almost limitless different classes of documents. In other words, XML allows creation of tag names that may provide additional information regarding what the element within a tag means. As such, tags are referred to as a field name which places a label on a field containing a unit of data. Labels within XML may be chosen by the programmer to be anything that makes sense in the context of a given application. Tags may also include attributes. Attributes are qualifiers on a tag that provide additional information about the element identified by the tag. Attributes may include an attribute name and an attribute value(s).

Requests received by the ApiService class 42 preferably include request parameters. The request parameters are formed to include a sequence of tags with corresponding units of data. An exemplary request parameter is a request name parameter to identify the nature, type or category of request received. Other exemplary request parameters may identify ranges of data, logic conditions or any other parameters associated with a request for data. The ApiService class 42 may translate the request parameters to generate an input message. In addition, responses